Guide de Fuite Game(V1.0)

Nom et Prénom : Alaa Otay

Classe : CPI1 TD2

Coté Conception :

Diagramme d’utilisation :

*) Acteurs :

- Joueur : Contrôle l'objet Alister pour éviter les monstres et collecter des points.
- Monstre : Chasse le joueur et provoque la fin du jeu s'il le touche.

*) Cas d'utilisation :

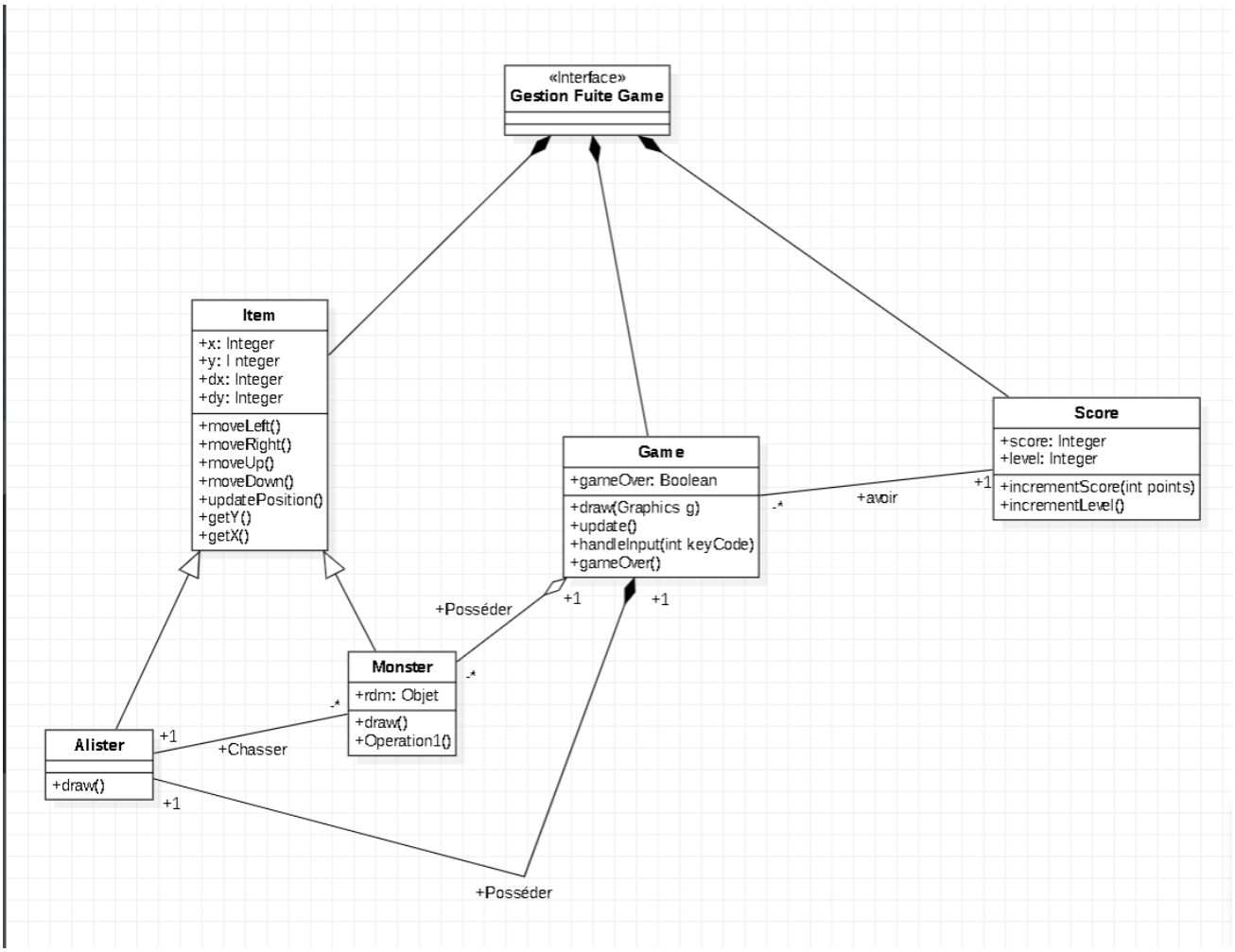
Joueur :

- Initialisation du jeu : décide la début du jeu par clique sur « Start »
- Bouger Alister : Le joueur peut déplacer Alister à gauche, à droite, en haut ou en bas pour éviter les monstres.
- Collecter des points : Le joueur gagne des points en évitant les monstres.
- Fin du jeu : Si Alister est touché par un monstre, le jeu se termine.
- Répéter jeu : Si le joueur souhaite rejouer, il peut cliquer sur le bouton "restart"

Monstre :

- Chasser Alister : Le monstre se déplace aléatoirement sur l'écran pour trouver Alister et le toucher.
- Collision avec les bords de l'écran : Si le monstre atteint le bord de l'écran, il change de direction pour continuer à chasser Alister.

Diagramme de Classes :



Coté Code:

Mon code formé de 5 classes Java :

*) Classe Item :

- getX() et getY():retournent les coordonnées actuelles de l'objet

-reverseDirection(): Cette méthode inverse la direction de l'objet

-updatePosition(): Cette méthode met à jour les coordonnées de l'objet aprés mouvement.

- moveLeft(), moveRight(), moveUp(), moveDown(): Ces méthodes déplacent horizontalement ou verticalement l'objet

*) Classe Alister :

- Ce code est une méthode de la classe qui dessine un polygone jaune en forme de "bonhomme de neige". La méthode prend un objet Graphics g en entrée qui est utilisé pour dessiner les formes sur l'écran.

-dans la methode Draw en utilisant biblio Graphics en a la méthode "g.fillPolygon" dessine le polygone en étoile en utilisant les tableaux "xpts" et "ypts" des points x2...x29 et y2...y29

-la méthode collidewithmonster() return un boolean si x et y de alister égale a monster.getX()+sz et monster.gety()+sz avec(sz:la moitié de la taille de l'objet, car on suppose que l'objet est un carré de côté 60).

- la méthode collideBorder() le jeu est dans un rectangle avec des coordonnées (0,0) pour le coin supérieur gauche et (570,570) pour le coin inférieur droit.

*) Classe Monster :

- draw :la méme avec draw de classe alister

- handleBorderCollision() : Cette méthode vérifie si le monstre a atteint les limites de l'écran

- hunting() : Cette méthode fait chasser le monstre de manière aléatoire

*) Classe Game panel :

- startTimer(): démarre le Timer si le jeu est en attente.

- reset(): réinitialise le jeu en créant de nouveaux personnages et ennemis, en remettant le score à zéro et en démarrant le Timer.

- keyPressed(KeyEvent e): détecte quelle touche a été pressée et appelle la méthode

- paint(Graphics g): dessine tous les éléments du jeu (personnage, ennemis, score, etc.), vérifie s'il y a eu collision avec les bords ou les ennemis. dans cette méthode j'ai choisie le nombre max monsters 6 et on ajoute un chaque 100 pts pour passer au niveau suivant

*) Classe FuiteGame :

(hérite de la classe "JFrame" et implémente l'interface "ActionListener")

- Elle définit une fenêtre de jeu avec des boutons pour les contrôles du jeu, ainsi qu'un panneau pour l'affichage du jeu lui-même.

- La méthode "init()" définit les différents éléments de la fenêtre de jeu, tels que le panneau de jeu et les boutons de contrôle. Elle ajoute également un label pour donner un conseil sur les contrôles du jeu .

Merci pour votre attention Monsieur 😊